

## دروس لغة الأسمبلي التابعة لموقع الفريق العربي للبرمجة <http://www.arabteam2000.com>

جميع الحقوق محفوظة للفريق العربي للبرمجة

يمنع منعاً باتاً مسح عنوان الموقع أو اسم المؤلف من هذه الدروس إلا بإذن صريح من إدارة موقع الفريق العربي للبرمجة

ملاحظة:

هذه الدروس هي بالأساس مقتبسة من منهاج السنة الثانية قسم هندسة الحاسبات بجامعة حلب

### الجزء الأول مقدمة في لغة الأسمبلي

## لمحة عن أنظمة العد

### تمهيد

اعتاد الإنسان على نظام العد العشري لأنه كان يملك عشرة أصابع في يديه، فعندما يريد إحصاء الأشياء أمامه فكان يقابل كل عنصر من الموجودات أمامه بإصبع واحدة من يديه، و عندما تنتهي أصابع يديه فإنه يحتاج إلى شخص آخر يرفع إصبع واحدة حيث تمثل كل إصبع من أصابع الشخص الثاني عشرة أصابع من أصابع الشخص الأول و بذلك كان الثاني يلعب دور العشرات أما الأول فيلعب دور الآحاد.

و بعد اختراع الكتابة سارع علماء الرياضيات إلى تحويل نظام العد العشري إلى صيغة كتابية، فاعتمدوا الأساس التالي: ( تمثل الأعداد من 1 حتى 9 برمز واحد فقط أما العدد الذي يأتي بعد التسعة فهو عبارة عن مزيج رمزين الأول هو الصفر و الثاني هو الواحد ).

من الفكرة السابقة نجد أن الرموز الأساسية لنظام العد العشري هي من الصفر حتى التسعة أي هي عشرة رموز نستطيع من خلالها تكوين عدد أي عدد طبيعي.

### طريقة العد:

نبدأ بالعد اعتباراً من أول رمز و هو الصفر و نزيد بمقدار واحد واحد إلى أن نصل إلى نهاية الرموز ألا و هو التسعة، و إذا أردنا المتابعة فإننا نصفر الخانة التي نتعامل معها و نضيف واحد إلى الخانة المجاورة لنحصل على الرقم عشرة ( 10 ) و من ثم نبدأ بزيادة الآحاد من جديد حتى نصل إلى 19 عندها نصفر الآحاد و نضيف واحد إلى خانة العشرات فينتج العدد 20 و هكذا حتى نصل إلى العدد 99 عندها نحاول زيادة خانة الآحاد فلا نستطيع فنصفرها و نحاول زيادة العشرات فلا نستطيع أيضاً فنصفرها و نزيد خانة إلى منزلة المئات فنحصل على العدد 100.

### العد بالنظام الست عشري

يختلف هذا النظام عن سلفه بأن الرموز الأساسية هي من الصفر حتى التسعة و يأتي بعد التسعة الأحرف من A حتى F أي أن الرموز الأساسية هي:

{ 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F }

و لكي نستطيع العد بسهولة في هذا النظام أعد قراءة التمهيد و لكن تخيل جداً أن للإنسان ست عشرة إصبع في كل يد ثمانية أصابع !!

تمرين على العد بالنظام الست عشري:

0,1,2,...,9,A,B,...,F,10,11,12,13,14,...,19,1A,1B,1C,...,1F,20,21,...,29,2A,2B,...,2F,30,...,99,9A,9B,...,9F,A0,A1,A2,...,A9,AA,AB,AC,...,AF,...,FF,100,...,119,11A,11B,...,199,19A,...

### نظام العد الثنائي

تتطلب أجهزة الحواسيب و الأجهزة الإلكترونية نظام عد جديد ملائم لطبيعة هذه الأجهزة، فنحن نعلم أن جميع الأجهزة الإلكترونية تعمل على التيار الكهربائي و الذي له حالتين هما الوضع on و الوضع off .

و بذلك كان النظام الثنائي هو الحل حيث اعتمد على رمزين فقط في تمثيل أعدداه هما الصفر و الواحد {0,1} .

العد بالنظام الثنائي:

0000,0001,0010,0011,0100,0101,0110,0111,1000,1001,1010,1011,1100,1101,1110,1111

### التحويل بين نظم الأعداد

يلزمنا في لغة الأسملي التحويلات التالية:

1 - التحويل من الثنائي إلى العشري.

2 - التحويل من الست عشري إلى العشري.

3 - التحويل من العشري إلى الثنائي.

و سنعطي مثلاً عن كل حالة من هذه الحالات:

مثال 1 : حول الرقم الثنائي التالي 0100 إلى مقابله في نظام العد العشري:

$$(0100)_b = 0 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 0 \times 2^3 = 0 + 0 + 4 + 0 = 4$$

مثال 2 : حول العدد الست عشري التالي 33A إلى مقابله في نظام العد العشري:

$$(33A)_h = 10 \times 16^0 + 3 \times 16^1 + 3 \times 16^2 = 10 + 48 + 768 = 826$$

مثال 3 : حول العدد العشري التالي 30 إلى مقابله في النظام الثنائي:

لدينا الجدول المرسوم جانباً:

...	128	64	32	16	8	4	2	1

نستخدم هذا الجدول من أجل هذا النوع من

التحويل فلتحويل العدد العشري 30 نلاحظ أنه

مكون من  $16+8+4+2$  فنضع وحدات تحت الأعداد 16 و 8 و 4 و 2 ونملأ الباقي أصفاراً، و بذلك نحصل على الرقم

الثنائي المقابل.

### المتعم الثنائي و كيفية الحصول عليه

يستخدم المتعم الثنائي من أجل تمثيل الأعداد السالبة في الحاسب في النظام الثنائي و لتمثيل عدد سالب نتبع الخطوات التالية:

1 - نكتب العدد بالنظام الثنائي.

2 - نقلب الأصفار و الواحدات أصفاراً.

3 - نضيف واحد إلى الرقم الناتج.

مثال: مثل العدد 30 - بالنظام الثنائي عن طريق المتعم الثنائي:

$$(30)_d = 0001\ 1110$$

$$\text{نقلب} \Rightarrow 1110\ 0001$$

$$\text{نضيف} \Rightarrow 1110\ 0010$$

## لمحة عن الحاسب

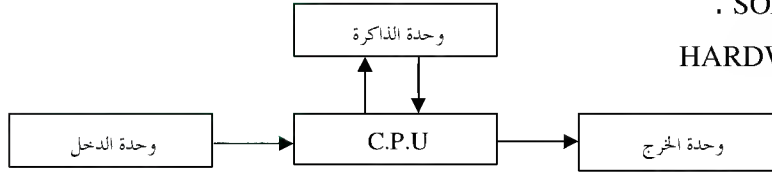
يُعرف الحاسب الرقمي بأنه نظام إلكتروني لمعالجة المعطيات، ويتألف من قسمين أساسيين:

القسم الأول : البرمجيات SOFTWARE .

القسم الثاني : الكيان الصلب HARDWARE

و يقسم الكيان الصلب إلى أقسام

رئيسية هي :



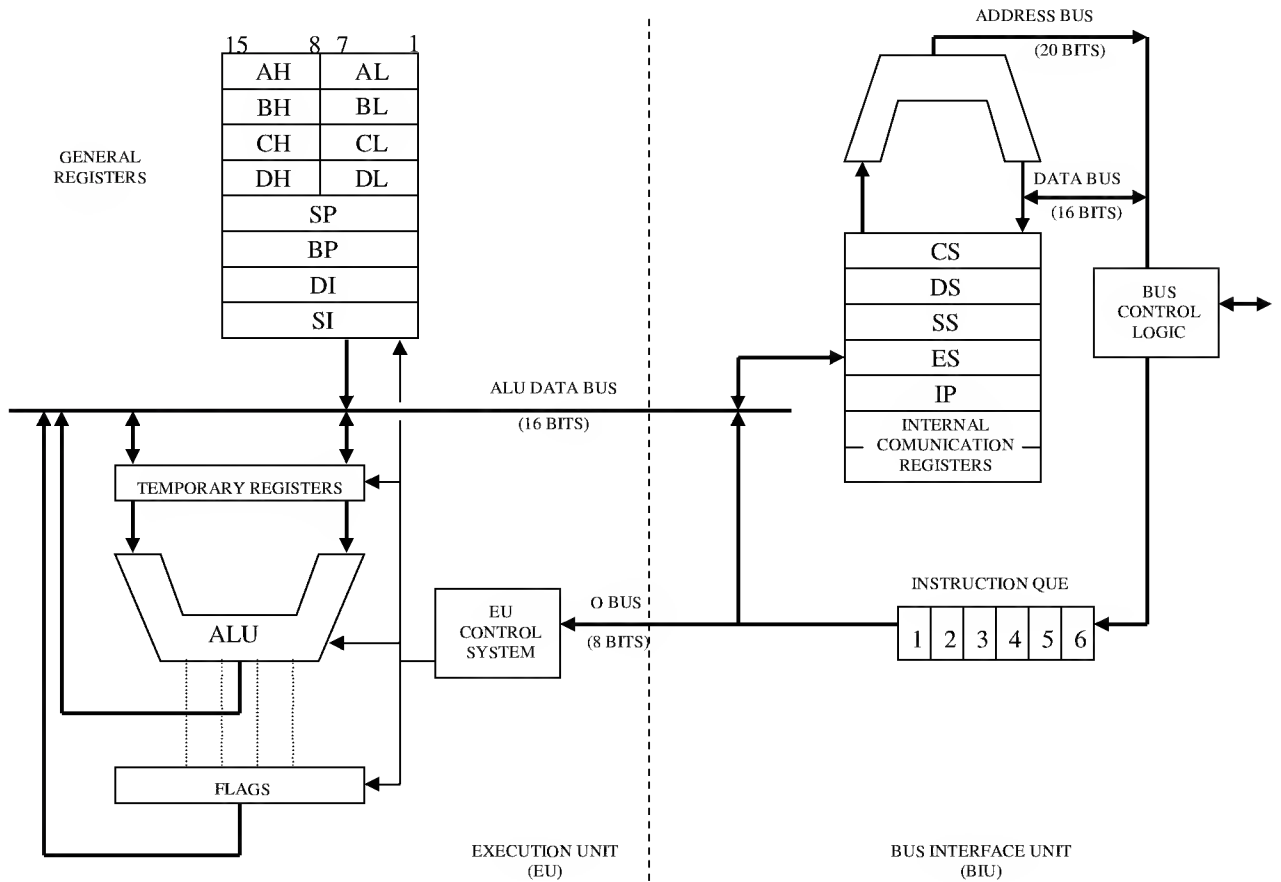
(1) وحدة الدخل: تتم من خلالها إدخال المعطيات الرقمية.

(2) وحدة الخرج: تتم من خلالها إظهار النتائج بعد معالجة المعطيات.

(3) وحدة المعالجة المركزية: هي المسؤولة عن العمليات الحسابية والمنطقية ومعالجة البيانات.

(4) وحدة الذاكرة: تخزن البرامج والمعطيات.

### البنية الداخلية للمعالج 8086



يتألف المعالج 8086 من وحدتين منفصلتين هما :

- (1) وحدة ملائمة الممرات ( Execution Unit ) : و سنرمز لها بالرمز EU .
  - (2) وحدة التنفيذ ( Bus interface Unit ) : و سنرمز لها بالرمز BIU .
- بشكل عام فإن الـ BIU مسؤولة عن معظم الأعمال مثل : إحضار التعليمات، قراءة و كتابة المتحولات في الذاكرة، إدخال و إخراج المعطيات من و إلى الأجهزة المحيطية.
- أما الـ EU فهي مسؤولة عن تنفيذ التعليمات. و كلا الوحدتين تعملان بشكل متوازٍ لتخفيض الزمن المطلوب لإحضار عدة تعليمات و تنفيذها.

ملاحظة: من الجدير بالذكر بأن هنالك ثلاثة ممرات في الحاسب و هي:

- (1) ممر المعطيات DATA BUS : و يصل بين المعالج و الذاكرة وظيفته نقل المعطيات من و إلى الذاكرة.
- (2) ممر العناوين ADDRESS BUS : و يصل بين المعالج و الذاكرة أيضاً و وظيفته نقل العناوين من المعالج إلى الذاكرة.
- (3) ممر التحكم CONTROL BUS : لتنسيق عمل الممرين السابقين.

#### وحدة ملائمة الممرات Bus Interface Unit

و تستخدم لملائمة المعالج مع العالم الخارجي. و تتألف من : جامع العناوين، مسجلات المقاطع، وحدة التحكم بالحرف، صف التعليمات.

تقوم وحدة الـ BIU بالتحكم بممر المعطيات و ممر العناوين و ممر التحكم .

تحضر BIU التعليمات من الذاكرة بايت بايت و تضعها فيما يسمى برتل التعليمات (صف التعليمات) الذي يتسع لست بايتات كحد أعظمي و من الطبيعي أن التعليمات التي تدخل رتل التعليمات أولاً يتم تنفيذها أولاً للمحافظة على ترتيب التعليمات و يدعى هذا المبدأ بـ الداخل أولاً خارج أولاً First In Last Out و نرمز لهذا المبدأ بـ FIFO.

إن إحضار شيفرة التعليمات التالية يتم عندما تكون وحدة التنفيذ EU مشغولة بتنفيذ التعليمات الحالية ( هذه إحدى محسنات المعالج 8086 عن أسلافه حيث كانت الـ CPU في المعالجات السابقة للمعالج 8086 تتوقف عن العمل خلال فترة تنفيذ التعليمات الحالية ).

عندما تفك وحدة التنفيذ EU شيفرة تعليمات ما من رتل التعليمات و تكون هذه التعليمات تعليمات تؤدي إلى تغيير تسلسل تعليمات البرنامج (قفز إلى برنامج فرعي مثلاً) عندها يتم تصفير رتل التعليمات و إعادة ملئه من جديد بتعليمات البرنامج الفرعي ( لأن وحدة ملائمة الممرات BIU تجلب التعليمات دون معرفة ما تؤديه هذه التعليمات).

ملاحظة: جامع العناوين و مسجلات المقاطع سيتم شرحها لاحقاً.

#### وحدة التنفيذ Execution Unit

و هي مسؤولة عن فك شيفرة التعليمات و تنفيذها و تتألف من :

- (1) وحدة الحساب و المنطق.
- (2) مسجل الأعلام.
- (3) ثمانية مسجلات للأغراض العامة.
- (4) مسجلات مؤقتة.

## 5) منطق التحكم بـ EU.

تجلب وحدة التنفيذ EU التعليمات من مقدمة رتل التعليمات في وحدة ملائمة الممرات BIU و تفك شيفرتها و تقوم بالعمل الذي تملبه كل تعليمة فإذا احتاجت هذه الوحدة ( EU ) إلى معلومة مخزنة في الذاكرة فإنها تأمر وحدة ملائمة الممرات BIU بإحضارها و ذلك عن طريق إعطائها عنوان هذه المعلومة في الذاكرة.

إن من أحد أهم وظائف EU هو تنفيذ العمليات الحسابية و المنطقية على المعلومات، و أثناء سير التنفيذ تقوم EU بفحص مسجل الأعلام بعد كل تعليمة ( مسجل الأعلام : هو عبارة عن ستة عشر بت تعبر عن حالة المعالج بعد تنفيذ كل تعليمة ) .

مسجلات الأغراض العامة هي ثمانية مسجلات طول كل مسجل منها 2 بايت و هذه المسجلات هي AX,BX,CX,DX,SI,DI,BP,SP .

## بنية الذاكرة

تتألف الذاكرة من حجرات متسلسلة سعة كل منها 8 بت (واحد بايت) ، ترقم هذه الحجرات من الصفر و حتى نهاية الذاكرة و يستخدم النظام الست عشري عادة في عملية الترقيم و بذلك يكون لكل حجرة رقم يميزها عن غيرها، يدعى هذا الرقم بعنوان تلك الحجرة.

يوضع داخل كل حجرة رقم ست عشري يتراوح بين 0 و FF و يدعى هذا الرقم بمحتوى تلك الحجرة.

يوجد بين المعالج و الذاكرة ممران هما ممر المعطيات بعرض 16 بت و ممر العناوين بعرض 20 بت.

فمثلاً عندما يحتاج المعالج إلى القيمة المخزنة في الحجرة ذات الرقم 100 ( عنوانها 100 ) فإن الرقم 100 يمثل بشكل ثنائي و يوضع على ممر العناوين و يرسل إلى الذاكرة، و حالما تستلم الذاكرة هذا العنوان فإن محتوى الحجرة 100 يرسل إلى المعالج عن طريق ممر المعطيات.

إن كون ممر العناوين ذو عرض 20 بت ( 20 خط نقل) هذا يعني أنه يستطيع نقل رقم ثنائي ذو 20 خانة أي أن أكبر قيمة يمكن وضعها على ممر العناوين هي :

$$2^{20} = 1048576 \approx 1MB$$

و بذلك يستطيع المعالج 8086 عنونة واحد ميغا من الذاكرة فقط.

## مقاطع الذاكرة (هذه الفقرة مرتبطة ارتباطاً وثيقاً بالمسجلات)

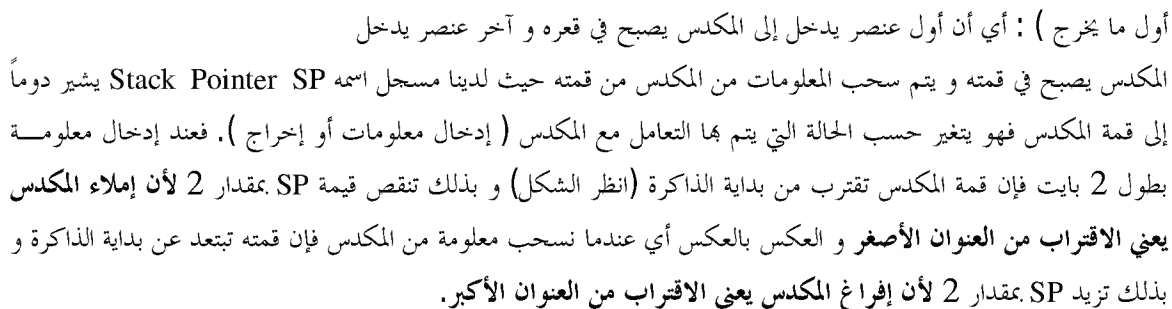
يتعامل المعالج كما ذكرنا مع واحد ميغا من الذاكرة، و يمكن أن نقتطع من هذه الميغا أربعة مقاطع أساسية يتعامل معها برنامجنا بشكل مباشر (أي أنه لا تتم الاستفادة من كل الذاكرة بآن واحد) و هذه المقاطع الأربعة هي:

### 1) مقطع الشيفرة Code Segment CS

يخصص هذا المقطع من الذاكرة -كما هو واضح من تسميته - لتخزين شيفرة البرنامج. و هناك مسجل له نفس الاسم CS موجود في المعالج يحتفظ بقيمة تدل على بداية هذا المقطع في الذاكرة و يساعده المسجل (Instruction Pointer) IP الذي يحتفظ بعنوان التعليمة التي ستنفذ الآن و تعدل قيمته آلياً ليشير إلى عنوان التعليمة التالية.

### 2) مقطع المعطيات Data Segment DS

### Stack Segment SS (3) مقطع المكسد



**Extra Segment ES (4) مقطع المعطيات الإضافي**

يستخدم عند الحاجة إلى استخدام مقطعي معطيات بنفس الوقت و بذلك نستطيع الاستفادة من مساحة أكبر في الذاكرة. و يساعده المسجل Destination Index DI الموجود في المعالج و الذي يشير إلى الإزاحة بالنسبة إلى بدايته.

ملاحظة: يجب التمييز بين المقطع و مسجل المقطع حيث المقطع هو جزء من الذاكرة بينما مسجل المقطع يتألف من بايتين و هو موجود في المعالج.

## Registers

يملك المعالج 8086 أربعة مجموعات من المسجلات ذات 16 بت يستطيع البرمج الوصول إليها وهي:

- (1) مؤشر التعليم IP
- (2) أربعة مسجلات معطيات AX,BX,CX,DX .
- (3) أربعة مسجلات تأشير و فهرسة SI,DI,BP,SP .
- (4) أربعة مسجلات مقاطع CS,DS,SS,ES .

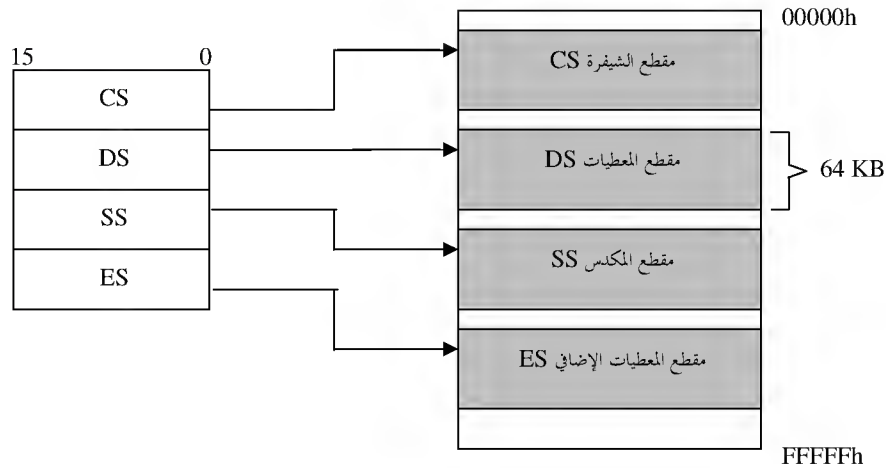
بالإضافة إلى ذلك يوجد مسجل آخر هو مسجل الأعلام و يدعى أيضاً مسجل الحالة و هو مسجل ذو 16 بت و لكن نستخدم منه 9 خانات فقط.

سنشرح كل من هذه المسجلات بالتفصيل :

### المجموعة الأولى : مسجلات المقاطع

و هي عبارة عن أربعة مسجلات طول كل منها 16 بت أي 2 بايت و هي :

- (1) مسجل مقطع الشيفرة CS : يحتوي على عنوان أول حجرة في مقطع شيفرة البرنامج في الذاكرة، أي أنه يشير إلى بداية مقطع الشيفرة.
- (2) مسجل مقطع المعطيات DS : يحتوي على عنوان أول حجرة في مقطع المعطيات في الذاكرة، أي أنه يشير إلى بداية مقطع المعطيات.
- (3) مسجل مقطع المكس SS : يحتوي على عنوان أول حجرة في مقطع المكس في الذاكرة، أي أنه يشير إلى بداية مقطع المكس.
- (4) مسجل مقطع المعطيات الإضافي ES : يحتوي على عنوان أول حجرة في مقطع المعطيات الإضافي في الذاكرة، أي أنه يشير إلى بداية مقطع المعطيات الإضافي.



### المجموعة الثانية: مسجلات الفهرسة و التأشير

و هي عبارة عن أربعة مسجلات مساعدة تساعد في إيجاد العنوان الفيزيائي بالتعاون مع مسجلات المقاطع، و طول هذه المسجلات 16 بت أي 2 بايت، و هي :

- (1) مسجل دليل المصدر Source Index SI : يخزن فيه عنوان يدل على الإزاحة ضمن مقطع المعطيات DS و بمعنى آخر يستعمل في إمساك العناوين الفعالة من أجل التعليمات التي تتناول المعطيات المخزنة في مقطع المعطيات في الذاكرة.
- (2) مسجل دليل الهدف Destination Index DI : يخزن فيه عنوان يدل على الإزاحة ضمن مقطع المعطيات الإضافي ES ، و بمعنى آخر يستعمل مسجل دليل الهدف DI من أجل استنتاج العنوان الفيزيائي الذي يحدد حجرة متحول الهدف.



(3) مسجل مؤشر المكس Stack Pointer SP : يسمح مؤشر المكس بوصول سهل للحجرات في مقطع المكس الموجود في الذاكرة حيث أن القيمة في SP تمثل العنوان الفعال لحجرة المكس التالية التي يمكن الوصول إليها نسبة إلى العنوان الحالي الموجود في مسجل مقطع المكس SS و يحتفظ SP دوماً بقيمة تدل على قمة المكس ، هذا و إن قيمة هذا المسجل تتعدل تلقائياً عند وضع أو سحب معلومة بالمكس.

(4) مسجل مؤشر القاعدة Base Pointer BP : يحوي قيمة تدل على الإزاحة بالنسبة لمقطع المكس SS و هو يستخدم لقراءة المعطيات ضمن مقطع المكس بدون إزالتها من المكس.

### المجموعة الثالثة: مسجلات المعطيات

تستخدم هذه المسجلات من أجل التخزين المؤقت للنتائج المرحلية أثناء تنفيذ البرنامج حيث أن تخزين المعطيات في هذه المسجلات يمكننا من الولوج إلى تلك المعطيات بشكل أسرع مما لو كانت في الذاكرة، و تقسم المسجلات إلى :

- (1) مسجل المراكم Accumulator و يرمز له بالرمز A .
- (2) مسجل القاعدة Base و يرمز له بالرمز B .
- (3) مسجل العد Count و يرمز له بالرمز C .
- (4) مسجل المعطيات Data و يرمز له بالرمز D .

و كل مسجل من المسجلات السابقة يمكن استعماله إما ككلمة 16 بت و يدل على ذلك بكتابة الحرف X بعد اسم المسجل أو يمكن استعماله كبائتين كل منهما 8 بت و يدل على ذلك باستخدام الحرفين H,L حيث :

L للبائت ذو العنوان الأصغر ، مثال AL .

H للبائت ذو العنوان الأكبر ، مثال BH .

هكذا و إن كلاً من هذه المسجلات يمكن استخدامه من أجل التعليمات الرياضية أو المنطقية في لغة التجميع مثل And, Add . و من أجل بعض التعليمات مثل البرامج التي تحتوي على تعليمات سلاسل فإنها تستعمل مسجلات معينة مثل استعمال المسجل C لتخزين العدد الذي يمثل عدد البائتين التي ستنفذ عليها تعليمات السلاسل ( عدد مرات تكرار تعليمة السلسلة )

### مسجل مؤشر التعليمة Instruction Pointer IP

هذا المسجل يحدد موقع التعليمة التالية التي ستنفذ في مقطع الشيفرة و بعد جلب شيفرة التعليمة من الذاكرة فإن BIU تعدل قيمة IP بحيث تشير إلى التعليمة التالية في الذاكرة ( التعديل يتم آلياً ) .

### مسجل الأعلام Flags Register

هو مسجل ذو 16 بت موجود في وحدة التنفيذ كما هو واضح بالشكل :

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				OF	DF	IF	TF	SF	ZF		AF		PF		CF

و كما نلاحظ من الشكل السابق أنه يوجد ستة أعلام للحالة هي CF, PF, AF, ZF, SF, OF ، و كذلك يوجد ثلاثة أعلام للتحكم DF, IF, TF .

(أ) أعلام الحالة

تشير إلى الحالات الناتجة كنتيجة لتنفيذ تعليمة منطقية أو رياضية حيث تكون إما في حالة واحد منطقي Set أو تكون في حالة صفر منطقي Reset ، و سنلخص فيما يلي عمل كل منها:

### أولاً: علم الإنزياح Carry Flag

يكون في حالة الواحد المنطقي إذا وجد انزياح خارجي ( حمل ) أو استعارة من أجل الخانة الأخيرة (البت الأخير) و ذلك أثناء تنفيذ التعليمات الرياضية.

و يكون في حالة الصفر المنطقي إذا لم يوجد حمل أو استعارة من أجل البت الأخير.

أمثلة:

#### أولاً: حالة الإنزياح

لاحظ بأن النتيجة لم تتسع في ثمانية بتات و إنما تحتاج إلى تسع بتات و تعبر عن ذلك بثمانية بتات و  $CF=1$  أي أنه لدينا في اليد واحد.

ببساطة: فمهما كبر العددين فإن تسعة بتات يمكن أن تستوعبها.

7	6	5	4	3	2	1	0	
1	1	0	0	0	1	1	0	
1	1	0	0	0	1	1	1	+
1	0	0	0	1	1	0	1	

→ CF=1

#### ثانياً: حالة الاستعارة

لاحظ بأن العدد الأول الممثل ثانياً أصغر من العدد الثاني الممثل ثانياً أيضاً ، لذلك فعند إجراء عملية الطرح و في مثالنا هذا نحيلنا بت تاسع فيه القيمة واحد (استعارة) و بالتالي فإن  $CF=1$  أي لدينا استعارة من أجل البت الأعلى رتبة.

7	6	5	4	3	2	1	0	
0	1	0	1	1	0	1	1	
1	1	1	1	1	0	0	0	-
0	1	1	0	0	0	1	1	

و في المثالين السابقين نطبق نفس الكلام من أجل 2 بايت و لكن الإنزياح الخارج و الاستعارة تكون من أجل البت الخامس عشر (الأخير).

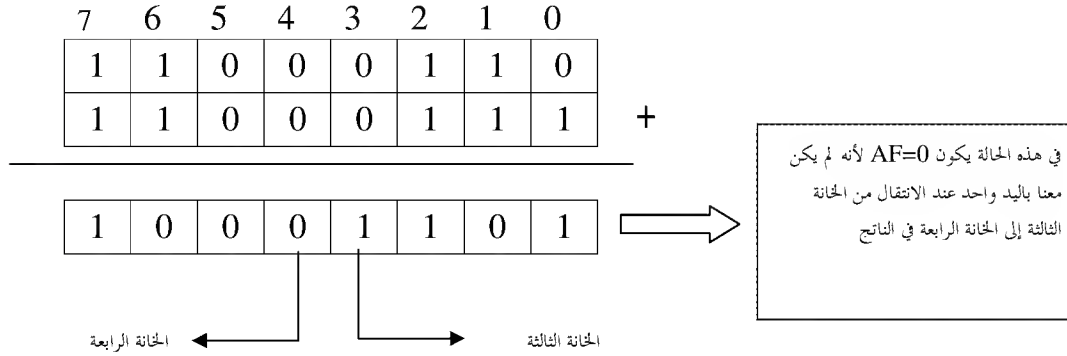
#### ثانياً: علم الازدواجية PF Parity Flag

يصبح في حالة واحد منطقي إذا كانت نتيجة آخر تعليمة تحوي على عدداً زوجياً من الخانات الواحدية ( بعد التحويل إلى النظام الثنائي طبعاً ) و إلا يكون في حالة الصفر المنطقي.

نلاحظ أن علم PF يفحص البايت السفلي فقط حتى لو كنا نتعامل مع كلمة ( 2 بايت ) ، أما عندما نتعامل مع بايت واحد فقط فإنه يفحصه كله.

#### ثالثاً: علم الإنزياح المساعد Auxiliary Flag AF

يكون في حالة الواحد المنطقي إذا وجد إنزياح من النصف السفلي إلى النصف العلوي أو استعارة من النصف العلوي إلى النصف السفلي و ذلك من أجل البايٲ السفلي من الكلمة ( 2 بايت ) و بمعنى آخر أنه إذا كان لدينا إنزياح من الخانة 3 إلى الخانة 4 فإن  $AF=1$  و ذلك في حال كانت المعطيات بايت واحد أو بايتين (كلمة)، و فيما عدا ذلك يكون  $AF=0$  .  
مثال:



#### رابعاً: علم الصفر Zero Flag ZF

يصبح في حالة واحد منطقي عندما يكون ناتج آخر عملية حسابية أو منطقية يساوي الصفر.  
يصبح في حالة صفر منطقي عندما يكون ناتج آخر عملية حسابية أو منطقية لا يساوي الصفر.

#### خامساً: علم الإشارة Sign Flag SF

يكون علم SF في حالة واحد منطقي Set إذا كانت نتيجة آخر عملية حسابية عدداً سالباً.  
يكون علم SF في حالة صفر منطقي Reset إذا كانت نتيجة آخر عملية حسابية عدداً موجباً.  
مصطلح: من إحدى طرق تمثيل الأعداد السالبة في الكمبيوتر هي اعتبار الخانة الأخيرة مخصصة للإشارة و بما أن البايٲ مكون من ثمانية خانات فسيتم اقتطاع الخانة الأخيرة منه من أجل الإشارة فإن احتوت على القيمة واحد فإن الخانات السبعة الباقية هي عدد ثنائي سالب أما إذا احتوت على القيمة صفر فإن الخانات السبعة المتبقية ما هي إلا عدد موجب.  
و بذلك يكون SF هو نسخة عن الخانة الأخيرة في الناتج عند اعتماد هذا النظام لتمثيل الأعداد السالبة.  
لاحظ أنه انطلاقاً من هذا المبدأ في التمثيل يمكننا تمثيل المجالات التالية من الأعداد:

من أجل بايت واحد من -128 إلى +127

من أجل بايتين من -32768 إلى +32767

#### سادساً: علم الطفحان Overflow Flag OF

يكون في حالة واحد منطقي عندما لا تتسع النتيجة في المكان المخصص لتخزينها أي تتجاوز القدرة التخزينية، أما إذا لم تكن النتيجة خارج المجال المحدد فإن OF يبقى في حالة الصفر المنطقي.

يحدث الطفحان في الحالات التالية:

(1) جمع أعداد موجبة كبيرة.

(2) جمع أعداد سالبة كبيرة.

(3) طرح عدد موجب كبير من عدد سالب كبير.

(4) طرح عدد سالب كبير من عدد موجب كبير.

ملاحظة: جميع الأعلام السابقة ما عدا CF تُقرأ فقط أي لا نستطيع تغيير محتواها لذلك يمكن قراءتها فقط و لا يمكن تغيير محتوياتها بواسطة تعليمات برمجية مباشرة.

المعالج مزود بتعليمات تستطيع اختبار حالة هذه الأعلام لتغيير تتابع تنفيذ البرنامج فمثلاً يمكن اختبار علم  $ZF=1$  كشرط من أجل القفز إلى جزء آخر من البرنامج.

و فيما يلي سنشرح أعلام التحكم:

### أولاً: علم الخطوة الوحيدة TF Trap Flag

يوضع بالحالة واحد منطقي عندما نرغب بتنفيذ البرنامج خطوة خطوة و هو مفيد عندما نريد تصحيح برنامجنا و استكشاف مواقع الأخطاء.

### ثانياً: علم المقاطعة IF Interrupt Flag

يستخدم من أجل التعبير عن إمكانية أو عدم إمكانية تنفيذ المقاطعة، فيوضع بالحالة واحد منطقي عندما لا نرغب بتنفيذ أي مقاطعة (المقاطعة محجوبة) أما عند وضعه في حالة الصفر المنطقي فإن المقاطعة مسموح بها. ملاحظة: المقاطعة هي عبارة عن خدمة تؤدي إلى عمل معين فمثلاً المقاطعة 21 و التي من أحد خدماتها العودة إلى نظام التشغيل.

### ثالثاً: علم الاتجاه DF Direction Flag

يدل على اتجاه سير العمليات التسلسلية.

عندما يكون في حالة واحد منطقي فإن السلسلة تكون من العنوان الأعلى إلى العنوان الأدنى.

عندما يكون في حالة صفر منطقي فإن السلسلة تكون من العنوان الأدنى إلى العنوان الأعلى.

## مفهوم العنوان الفيزيائي و الإزاحات

### مقدمة

لاحظنا أن الذاكرة بطول 1 ميغا بايت أي أنها مرقمة من 00000h إلى FFFFFh لذلك فإننا نحتاج أثناء عنوان المقاطع إلى رقم ست عشري بطول 20 بت ذلك لأن تمثيل رقم ست عشري بطول خمس خانات ( و هو المستخدم في ترقيم حجرات الذاكرة ) يحتاج إلى عشرين بت لكن مسجلات المقاطع و التي نستخدمها في العنوان هي بطول 16 بت فقط الأمر الذي يضطرنا إلى استنتاج عنوان فيزيائي بعشرين بت !!

### آلية الحصول على العنوان الفيزيائي PA Physical Address

يلزمنا لإيجاد العنوان الفيزيائي قيمتين هما :

- (1) قيمة مسجل المقطع
- (2) قيمة المسجل المساعد له

### فكرة Very good Tip :

عندما نريد إزاحة رقم ممثل بالنظام العشري خانة واحدة نحو اليسار فإننا نضربه بعشرة !!

مثال: هل تستطيع إزاحة الرقم 192 إلى اليسار خطوة واحدة ليصبح 1920 ؟؟

نعم و ذلك بضربه بعشرة كالتالي  $192 \times 10 = 1920$   
و كذلك الأمر في النظام الست عشري، فعندما نريد إزاحة رقم ست عشري فإننا نضربه بعشرة النظام الست عشري و التي هي

$$10h = 16d$$

↓                      ↓  
عشرة النظام الست عشري                      مقابلها في النظام العشري

لذلك يتم الحصول على العنوان الفيزيائي بالطريقة التالية:

- (1) نأخذ قيمة مسجل المقطع الممثلة بالنظام الست عشري و نضربها بعشرة النظام الست عشري فتتأخر قيمة مسجل المقطع خانة واحدة نحو اليسار.
- (2) نجمع قيمة المسجل المساعد لنفس المقطع و الممثلة أيضاً بالنظام الست عشري فتكون النتيجة هي حصولنا على العنوان الفيزيائي

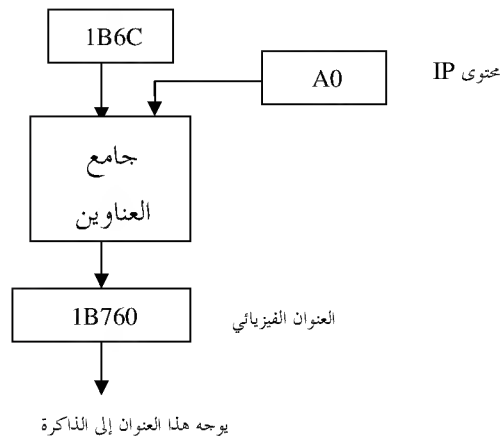
$$PA \text{ (Physical Address)} = \text{قيمة المسجل المساعد} + (10h \times \text{مسجل المقطع})$$

أمثلة:

بفرض لدينا مسجل مقطع الشيفرة CS يحتوي على 1B6C و قيمة مسجل مؤشر التعليم IP المساعد له هي A0 أوجد العنوان الفيزيائي للتعليم:

الحل:

$$PA = (CS \times 10h) + IP = 1B6C \times 10h + A0 = 1B760$$



مثال آخر: أوجد PA بفرض  $DS = 1000h$  و  $SI = 1F$ .

الحل:

$$PA = (1000 \times 10) + 1F = 1001F$$

الطريقة العكسية ( هذه الطريقة يجب إتقانها ذهنياً )

عندما تُعطى العنوان الفيزيائي و نريد استنتاج قيمة مسجل المقطع ( عنوان المقطع ) و قيمة المسجل المساعد له ( الإزاحة ) نتبع إحدى الطريقتين التاليتين :

الطريقة الأولى

- 1 - نأخذ الخانات الأربعة اليمينية من العنوان الفيزيائي المعطى و نعتبرها إزاحة ( أي نضع قيمتها في المسجل المساعد ) .
  - 2 - نصفر الخانات الأربعة الأولى من العنوان الفيزيائي فينتج معنا رقم ست عشري أول أربع خانات منه أصفاراً .
  - 3 - نحذف الصفر الأول من الرقم الناتج فينتج معنا رقم ست عشري هو قيمة مسجل المقطع .
- مثال:

بفرض لدينا عدد موجود في العنوان الفيزيائي 41000h أوجد قيمة مسجل المعطيات DS و قيمة المسجل المساعد له SI .  
الحل: حسب الطريقة بأخذ الخانات الأربعة الأولى من على اليمين تكون قيمة SI تساوي 1000h و هي الإزاحة.

$$DS = 4000h \quad (2-3)$$

طريقه أخرى

- 1 - نأخذ الخانة الأولى من العنوان الفيزيائي ونعتبرها إزاحة.
- 2 - نحذف تلك الخانة من العنوان الفيزيائي فيصبح الرقم الناتج مؤلف من أربع خانات و هو يمثل قيمة مسجل المقطع.

$$PA = 41000h \text{ كان يفرض كان}$$

الحل : بأخذ الخانة الأولى

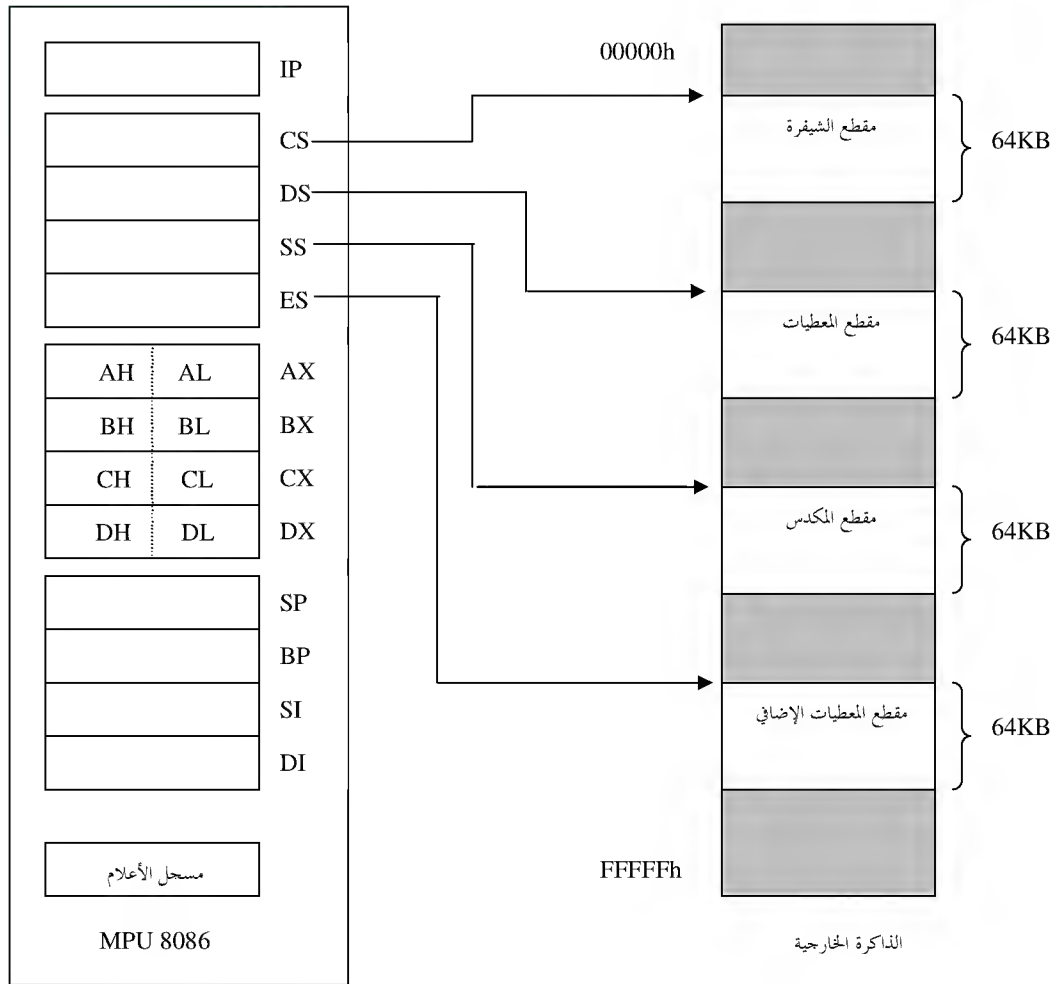
- 1) SI=0
- 2) DS=4100 أخذنا الخانات المتبقية من الرقم

أي أن

$$4100:0000 \equiv 4000:1000$$

$$\text{إزاحته عنوان} \equiv \text{إزاحته عنوان}$$

## الموديل البرمجي للمعالج 8086



## أساليب العنوان

### مقدمة:

إن حيز الذاكرة منظم على شكل بايتات معنونة من 00000h إلى FFFFFh لذلك من أجل كلمات المعطيات 16 بت يتم تخزين البايت السفلي في العنوان الأصغر و البايت العلوي في العنوان الأكبر كما نعلم أن الذاكرة تحتوي أربع مقاطع كل منها 64KB و هي مقطع الشيفرة و مقطع المعطيات و مقطع المكس و مقطع المعطيات الإضافي، حيث يتم الرجوع إلى هذه المقاطع بمساعدة مسجلات المقاطع ذات الـ 16 بت و هي CS, DS, SS, ES و كل من هذه المسجلات يحتوي عنواناً قاعدياً ذا 16 بت و الذي يستخدم في توليد العنوان الفيزيائي للذاكرة و الذي يشير إلى بداية المقطع المطابق في الذاكرة. يستطيع المبرمج تغيير القيم في مسجلات المقاطع برمجياً، فمثلاً: يمكن تهيئة مقطع معطيات جديد ببساطة و ذلك بتغيير قيمة المسجل DS عن طريق تنفيذ التعليمتين التاليتين:

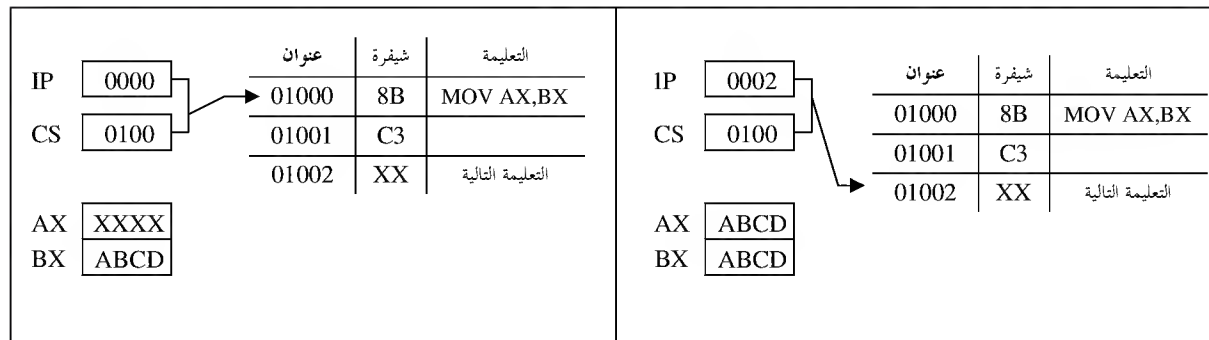
```
Mov AX,A000
```

```
Mov DS,AX
```

و سبب وجود هاتين التعليمتين هو عدم وجود تعليمة واحدة لتحميل مسجل مقطع بعدد ثابت. إن المعالج 8086 مزود بتسعة أنظمة عنوان مختلفة، و هي: العنوان بالمسجل - العنوان الفورية - العنوان المباشرة - العنوان غير المباشرة بالمسجل - العنوان القاعدية - العنوان المفهرسة - العنوان القاعدية المفهرسة - العنوان بالسلسلة - العنوان بالنافذة. و هذه الأنظمة التسعة عدا العنوان بالمسجل و العنوان الفورية تتطلب الرجوع إلى المتحول المخزن في الذاكرة لذلك نحتاج لأن تبدأ وحدة ملائمة الممرات BIU بدورة ممر لقراءة أو كتابة في الذاكرة و هكذا فإن كل نظام عنوان له طريقة مختلفة لحساب عنوان المتحول الذي سيخرج على ممر العناوين أثناء دورة الممر، و سندرس الآن كلاً من هذه الأنظمة بالتفصيل: ملاحظة: جميع التعليمات ستشرح لاحقاً.

### أولاً: نظام العنوان بالمسجل

في هذا النظام يكمن المتحول بمسجل داخلي للمعالج، فمثلاً التعليمة التي تستعمل نظام العنوان بالمسجل هي MOV AX,BX و التي تعني نقل محتوى BX (متحول المصدر) إلى المسجل AX (متحول الهدف) أي أن تنفيذ هذه التعليمة يتم دون الرجوع إلى الذاكرة أي في المعالج لأن كلا المسجلين AX و BX موجودين في المعالج:



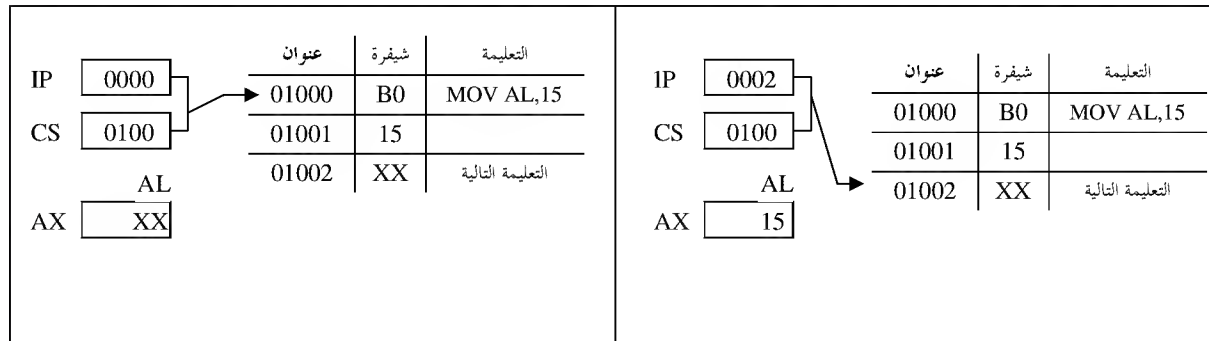
نلاحظ من الشكلين السابقين و في الشكل الأول نجد أنه قد تم توليد العنوان الفيزيائي للتعليمة بواسطة الـ IP و الـ CS حيث يتم إحضار التعليمة إلى المعالج و تتم فك شيفرتها (8BC3 من الجدول).



### ثانياً: نظام العنوان الفورية

في هذا النظام يكون المتحول جزء من التعليمة و ليس مضمون سجل أو عنوان حجرة ذاكرة حيث يدعى هذا المتحول بالمتحول الفوري و المتحولات الفورية تمثل معطيات ثابتة يمكن أن تكون بايت أو كلمة ( 2 بايت ).

مثال: MOV AL,15 نجد أن متحول المصدر هو 15h و هو متحول مصدر فوري ذو بايت واحد و الشكلان التاليان يوضحان حالة المعالج قبل و بعد تنفيذ التعليمة السابقة.



### ثالثاً: نظام العنوان المباشرة

يختلف هذا النظام عن نظام العنوان الفورية بأن الحجرات التي تلي رمز التعليمة تحوي على العنوان الفعال للذاكرة EA = Effective memory Address أي الإزاحة و هذا العنوان مؤلف من 16 بت حيث يتم توليد العنوان الفيزيائي انطلاقاً من DS و ES .

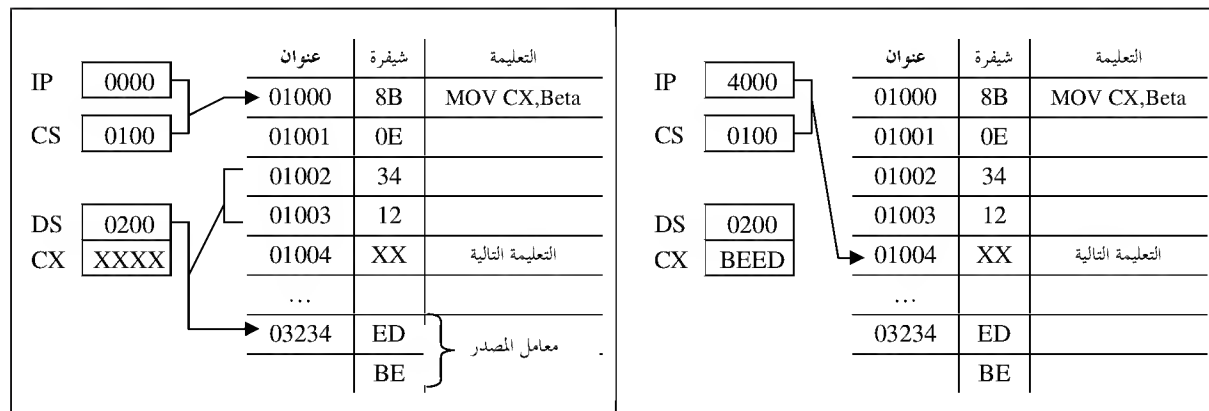
مثال:

MOV CX,[1234]

بفرض كان DS = 200 عندئذ العنوان الفيزيائي يحسب بالعلاقة :

$$PA = 200 \times 10h + 1134 = 03243h$$

ثم يذهب المعالج إلى الموقع 03243h في الذاكرة و يأخذ محتوى تلك الحجرة و يضعها في CL و يأخذ محتوى الحجرة التي تليها و يضعها في CH .

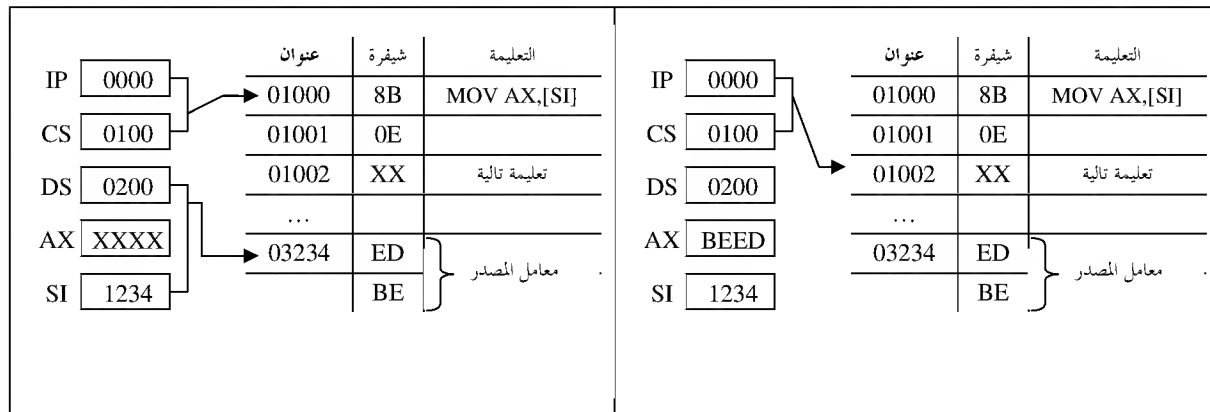


**رابعاً: نظام العنوان غير المباشرة بالمسجل :**

هذا النظام يشبه نظام العنوان المباشرة لكن يختلف عنه بأن العنوان الفعال ( إزاحة ) يكمن في مسجل مؤشر BX,BP أو مسجل دليل SI,DI .

مثال:

إن التعليمة التي تستخدم نظام العنوان غير المباشرة بالمسجل هي MOV AX,[SI] حيث يتم توليد العنوان الفيزيائي للمتحويل بالاستناد إلى SI و DS . عن طريق العلاقة  $PA = (DS \times 10h) + SI$  ، و يفرض كانت  $SI = 1234$  و  $DS = 200$  فإن  $PA = (0200 \times 10) + 1234 = 03234$  و هو معامل المصدر حيث يذهب المعالج إلى الحجرة 03234 و يأخذ منها قيمتها و يضعها في AL أما قيمة الحجرة التي تليها فيتم وضعها في AH و يبين الشكل التالي حالة المعالج قبل و بعد تنفيذ التعليمة السابقة:

**خامساً: نظام العنوان القاعدية**

في هذا النظام من العنوانية يحسب العنوان بواسطة جمع الإزاحة ( disp ) مع محتويات إما مسجل القاعدة BX أو مسجل مؤشر القاعدة BP مع القيمة الحالية الموجودة في المسجل DS أو SS على الترتيب أي:

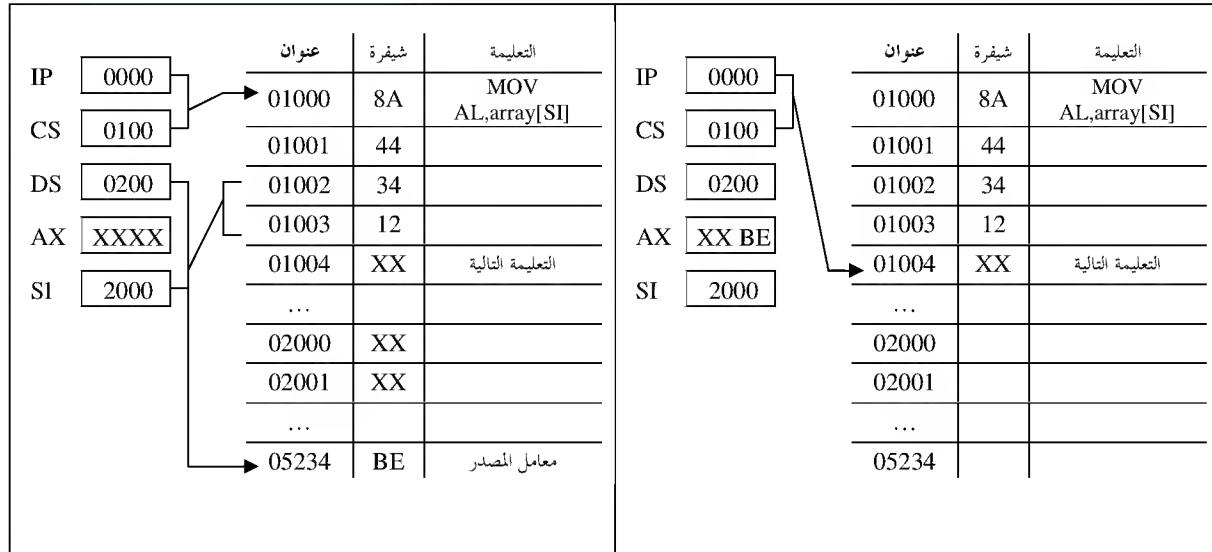
$$PA = (DS \times 10h) + BX + disp = (SS \times 10h) + BP + disp$$

إن تعليمة MOV التي تستخدم العنوان القاعدية لتحديد حجرة متحول الهدف هي

MOV [BX].Beta,AL

إن شيفرة التعليمة السابقة هي 3412 8870 و إن هذه التعليمة تستخدم مسجل القاعدة BX و الإزاحة المباشرة Beta لاشتقاق العنوان الفعال لمتحول الهدف حيث يتم تحقيق نظام العنوان القاعدية بواسطة تخصيص مسجل القاعدة أو مسجل مؤشر القاعدة بقوسين متوسطين ( مربعين ) متبوعاً بنقطة و إزاحة مباشرة ( Beta ) . إن متحول المصدر في هذه التعليمة متوضع في البايث السفلي من المراكم أي في AL و يفرض أن قيمة Beta هي 1234h فإن العنوان الفيزيائي لمتحول الهدف يتم حسابه بالعلاقة:

$$PA = (DS \times 10h) + BX + disp = 02000 + 1000 + 1234 = 04234h$$



هذا العنوان الفيزيائي تحسبه الـ BIU و من ثم تطلب الـ EU بدء دورة ممر كتابة في الذاكرة و هكذا يكتب متحول المصدر AL في حجرة الذاكرة ذات العنوان الفيزيائي 04234h أي بعد تنفيذ التعليمة تصبح حالة المعالج كما هو واضح في الشكل السابق.

#### سادساً: نظام العنوانية المفهرسة

في هذه الطريقة من العنوانية يتم الحصول على العنوان الفعال نتيجة جمع محتوى مسجل الفهرس إما DI أو SI إلى عنوان الإزاحة ( displacement ) و هذا النوع من العنوانية يناسب أغراض الجداول حيث يكون عنوان الإزاحة في بداية أول عنوان من الجدول و مسجل الفهرس يؤشر إلى أي عنصر من محتويات الجدول.

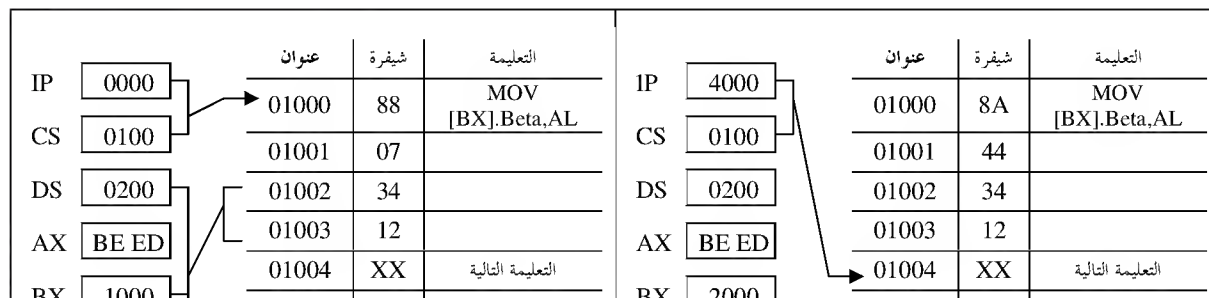
مثال: ليكن لدينا التعليمة التالية و التي شيفرتها 8A443412 و هي MOV AL,array[SI] .

هذه التعليمة يتم فيها تحديد متحول المصدر بواسطة العنوانية المفهرسة المباشرة حيث أن array تمثل الإزاحة المباشرة و هي تسبق مسجل الدليل الموجود ضمن قوسين متوسطين، حيث يتم توليد العنوان الفيزيائي التالي:

$$PA = ( DS \times 10h ) + EA$$

$$; EA = ( SI ) + disp \Rightarrow EA = 2000 + 1234 = 3234h$$

$$PA = ( DS \times 10h ) + EA = 02000 + 3234 = 05234h$$



إن نتيجة تنفيذ هذه التعليمات هي أن محتويات حجرة الذاكرة ذات العنوان الفيزيائي 05234h تنقل إلى AL و تصبح حالة المعالج كما هو موضح في الشكل السابق.

#### سابعاً: نظام العنوان القاعدية المفهرسة :

في هذا النوع من العنوان يتم الحصول على العنوان الفعال نتيجة جمع محتوى مسجل القاعدة مع مسجل الفهرس و في حال وجود عنوان يجب إضافة هذا العنوان إلى محتوى المسجلين المذكورين.

مثال: التعليمات MOV AH,[BX].Beta[SI] يتم فيها حساب العنوان الفعال لمتحول المصدر كما يلي:

$$EA = BX + Beta + SI$$

$$PA = ( DS \times 10h ) + EA = 02000 + 4234 = 06234h$$

و يبين الشكل التالي حالة المعالج قبل و بعد تنفيذ التعليمات:

	عنوان	شيفرة	التعليمات
IP	0000		
CS	0100		
DS	0200		
AX	XX XX		
BX	1000		
SI	2000		
	01000	8A	MOV AH ,[BX].Beta[SI]
	01001	20	
	01002	34	
	01003	12	
	01004	XX	التعليمات التالية
	...		
	02000	XX	
	02001	XX	
	...		
	06234	BE	معامل المصدر

	عنوان	شيفرة	التعليمات
IP	0000		
CS	0100		
DS	0200		
AX	BE XX		
BX	I000		
SI	2000		
	01000	8A	MOV AH ,[BX].Beta[SI]
	01001	20	
	01002	34	
	01003	12	
	01004	XX	التعليمات التالية
	...		
	02000	XX	
	02001	XX	
	...		
	06234	BE	معامل المصدر

حيث نلاحظ أنه بعد تنفيذ التعليمات تصبح محتويات AH = BEh والتي تمثل محتويات حجرة الذاكرة ذات العنوان الفيزيائي 06234h .

### ثامناً: نظام العنوان بالسلسلة

إن تعليمات السلسلة في مجموعة تعليمات المعالج 8086 تستعمل أوتوماتيكياً مسجل دليل المصدر و مسجل دليل الهدف لتعيين عناوين الفعالة لمتحولي المصدر و الهدف. فمثلاً تعليمة MOVS هي تعليمة النقل للسلسلة، و هي تستخدم SI و المقطع DS من أجل متحول المصدر و DI و المقطع ES من أجل متحول الهدف. و نلاحظ أنه لا SI و لا DI تظهران في تعليمة السلسلة.

### تاسعاً: نظام العنوان بالنافذة

يستعمل هذا النظام مع تعليمات الإدخال و الإخراج لنوافذ I/O . من أجل النوافذ في حيز عنوان I/O يستخدم فقط نظام العنوان المباشرة و نظام العنوان غير المباشرة لاستعمال المسجل DX . فمثلاً العنوان المباشرة لنافذة دخل تكون كما في التعليمة التالية:

IN AL,15h    Ū    IN AL,[15h]

تعني هذه التعليمة إدخال معطيات ذات بايت واحد من نافذة الدخل ذات العنوان 15h من حيز عنوان I/O إلى المسجل AL.

مثال آخر عن استعمال العنوان غير المباشرة للنافذة من أجل متحول المصدر هو التعليمة التالية:

IN AL,[DX]

هذا يعني إدخال معطيات ذات بايت واحد من نافذة الدخل التي عنوانها يكون محدد بواسطة مضمون مسجل DX فمثلاً: إذا

كان  $DX = 1234h$  فإن محتويات النافذة ذات العنوان 1234h يتم تحميلها في المسجل AL.